BIG DATA

PARK SANG HEE

INDEX

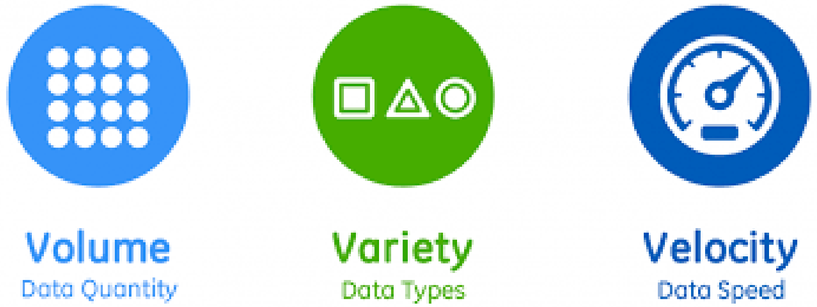
INDEX

Ⅰ. THEORY

빅데이터란?

빅 데이터는 통상적으로 사용되는 데이터 수집, 관리 및 처리 소프트웨어의 **수용 한계를 넘어서는 크기의 데이터**를 말한다. 빅 데이터의 사이즈는 단일 데이터 집합의 크기가 수십 테라바이트에서 수 페타바이트에 이르며, 그 크기가 끊임없이 변화하는 것이 특징이다.

빅데이터의 특징은 "3V" 로 설명할 수 있다. **데이터의 양(VOLUME), 데이터의 생성 속도(VELOCITY), 데이터 형태의 다양성(VARIETY)**으로 표현한다.



Ⅰ. THEORY

데이터의 정의

Data는 라틴어 단어 Datum의 복수형인 Data에서 유래했으며 라틴어에서 Datum의 뜻은 "present/gift, that which is given, debit" 이다. 현재에서도 기본적으로는 복수형 취급을 하나 가끔 하나의 고유명사화가 되어서 단수로 취급하는 경우도 있다.

1. 이론을 세우는 데 기초가 되는 사실. 또는 바탕이 되는 **자료**
2. 관찰이나 실험, 조사로 얻은 자실이나 **자료**
3. 컴퓨터가 처리할 수 있는 문자, 숫자, 소리, 그림 따위의 형태로 된 **자료**

정보가 아니라 자료임에 유의하자. **자료(data)를 가공해 얻는 것이 정보(information)**다.

Ⅰ. THEORY

암묵지와 형식지

암묵지는 “R사용법”, “자전거 타기” 등과 같이 학습과 체험을 통해 습득이 되는 무형의 지식을 말하고, **형식지**는 교과서, 매뉴얼, DB처럼 형상화된 지식이다. 암묵지와 형식지는 지식을 공유하고 발전시키면서 **상호작용**을 한다.

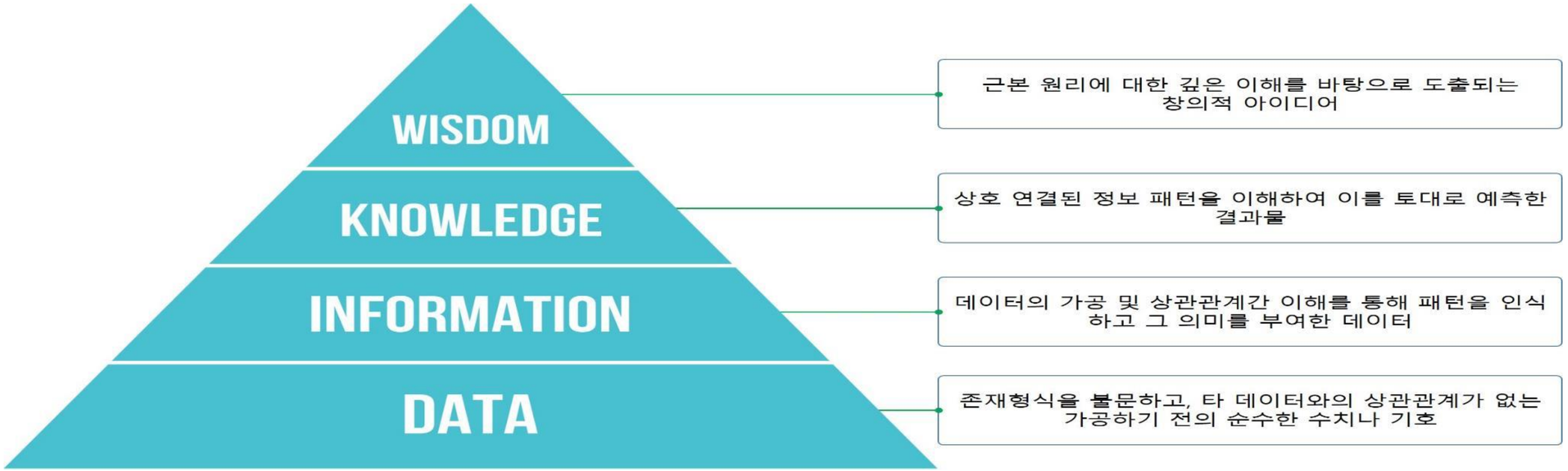
그림 : 지식 변환의 네가지 과정

상호작용	설명
사회화	암묵지에서 구체적인 개념 도출
내부화	형식지의 완성도를 높여 지식 체계로 전환
조합화	형식지를 학습하여 구체화된 개인지식으로 흡수
외부화	경험 공유를 통한 새로운 암묵지 창조

Ⅰ. THEORY

데이터와 정보

데이터는 객관적인 사실을 말하며, **정보**는 데이터를 통해서 의미가 도출된 것이며, **지식**은 다양한 정보를 통해 나온 유의미한 결과물이며, **지혜**는 지식의 축적과 아이디어가 결합된 창의적 산물이다.



Ⅰ. THEORY

데이터베이스(DB, DATABASE)

문자, 기호, 음성, 화상, 영상 등 상호 관련된 다수의 콘텐츠를 정보 처리 및 정보통신 기기에 의하여 체계적으로 수집·축적하여 다양한 용도와 방법으로 이용할 수 있도록 정리한 **정보의 집합체**

일반적으로 **DB**는 DATABASE를 뜻하며, **DBMS**는 DATA BASE MANAGEMENT SYSTEM을 뜻한다. DB는 말 그래도 데이터를 모아둔 것을 말하며, DBMS는 그 모아둔 데이터를 관리하는 시스템을 말한다.

통합된 데이터 : 동일한 내용의 데이터는 존재하지 않는다.
저장된 데이터 : 컴퓨터가 접근할 수 있는 저장 매체에 저장된다.
공용 데이터 : 여러 사용자가 서로 다른 목적으로 이용한다.
변화되는 데이터 : 갱신, 삭제, 삽입 등으로 데이터가 변화된다.

Ⅰ. THEORY

빅데이터의 출현 배경

빅데이터는 어느 날 갑자기 나타난 것이 아니라, 기존의 **데이터 처리 방식에서 한계**를 느낀 사람들이 처리 방식과 다루는 사람의 변화, 조직 차원에서 일어나는 변화를 지칭한다.

개별 기업의 고객 데이터 축적 및 활용 증가, 인터넷 확산, 저장 기술의 발전과 가격 하락, 모바일 시대의 도래와 스마트 단말의 보급, 클라우드 컴퓨팅 기술 발전, SNS와 사물 네트워크 확산 등이 맞물려 데이터 생산이 폭발적으로 증가하면서 **빅데이터 시대가 도래**하였다.

Ⅰ. THEORY

빅데이터의 핵심 기술 “클라우드 컴퓨팅”

클라우드 컴퓨팅(Cloud Computing)이란 **정보처리를 자신의 컴퓨터가 아닌 인터넷으로 연결된 다른 컴퓨터로 처리하는 기술**을 말한다. 우리가 사용하고 있는 개인용 컴퓨터(PC)에는 필요에 따라 구매한 소프트웨어가 설치되어 있고 동영상과 문서와 같은 데이터도 저장되어 있다. 문서를 작성하려면 자신의 컴퓨터에 저장되어 있는 글 과 같은 프로그램을 구동시켜야 한다.그러나 클라우드 컴퓨팅은 프로그램과 문서를 다른 곳에 저장해 놓고 내 컴퓨터로 그곳에 인터넷을 통해 접속해서 이용하는 방식이다. **자동차를 사지 않고 필요할 때 빌려서 쓰거나 대중교통을 이용하는 것**과 같다.

Ⅰ. THEORY

빅데이터의 병렬 분산 처리의 핵심 “맵리듀스”

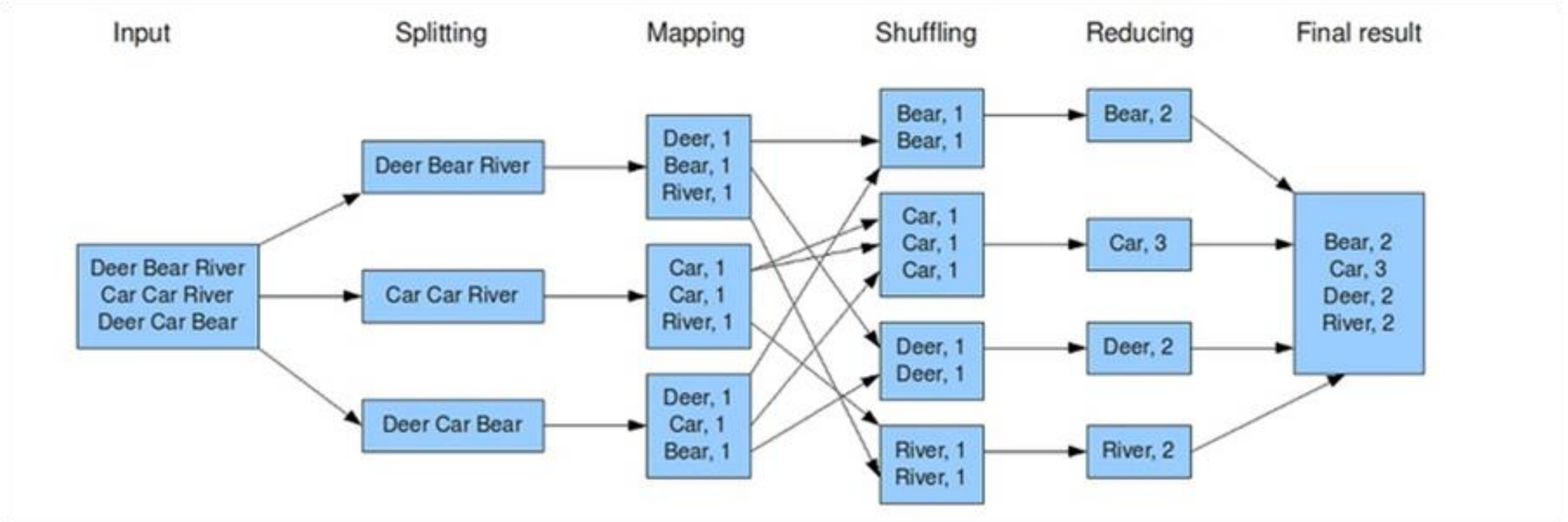
맵리듀스(MapReduce)는 **구글에서 대용량 데이터 처리를 분산 병렬 컴퓨팅에서 처리하기 위한 목적으로 제작**하여 2004년 발표한 소프트웨어 프레임워크다. 이 프레임워크는 페타바이트 이상의 대용량 데이터를 신뢰도가 낮은 컴퓨터로 구성된 클러스터 환경에서 병렬 처리를 지원하기 위해서 개발되었다. 이 프레임워크는 함수형 프로그래밍에서 일반적으로 사용되는 Map과 Reduce라는 함수 기반으로 주로 구성된다.

현재 MapReduce는 Java와 C++, 그리고 기타 언어에서 전용의 가능하도록 잘설되었다. 대표적으로 **아파치 하둡**에서 오픈 소스 소프트웨어로 적용되었다.

MapReduce

Ⅰ. THEORY

빅데이터의 병렬 분산 처리의 핵심 “맵리듀스”



[맵리듀스 처리 방법]

Ⅰ. THEORY

빅데이터가 만들어낸 본질적인 변화

“사전처리에서 사후처리 시대로”

전통적인 통계방법에서는 **실험계획이나 설계, 계획 등의 과정을 거쳐 사전 처리**를 한 다음 데이터를 수집한다. 이렇게 데이터를 수집하는 이유는 데이터를 수집하는 데 비용과 시간이 많이 들었기 때문이다. 그러나 산업혁명을 기점으로 데이터가 쏟아지는 양이 매우 커지고, 데이터를 수집하는 기술이 상당히 발전함에 따라 일단 **데이터를 모으고 그 안에서 필요한 데이터를 골라서 쓰는 사후 처리** 방식으로 바뀌게 되었다.

예를 들면 영화 관람객을 예측하는 모델을 만든다고 가정했을 때 사전 처리 방식은 모델 생성에 필요한 변수(감독, 장르, 배급사 등)들을 사전에 계획하고 그 변수들만 수집하지만, 사후 처리 방식은 실시간 SNS 관람객 후기, 실시간 예매율 등과 같이 실시간으로 발생하는 데이터를 모은 다음 처리하는 방식이다.

Ⅰ. THEORY

빅데이터가 만들어낸 본질적인 변화

“표본조사에서 전수조사로”

표본조사는 일반적으로 모집단의 특성을 파악하기 위해 모집단의 규모가 너무 커서 전수조사하기 어려울 때 확률추출로 표본을 뽑아 실시하는 조사이다.

빅데이터에서는 표본조사 전수조사의 개념이 사실상 무의미하다. 빅데이터의 목적은 어떤 집단의 특성 즉, 평균, 중앙값, 비율 등의 집단을 대표하는 특성값을 알고자 함이 아니라 개별 데이터에 대한 예측이 주 목적이기 때문에 대표성 보다는 개별 특성의 데이터가 많이 요구된다. 따라서 **조사하고자 하는 모집단에서 확률추출한 표본을 조사하기 보다는 모집단에서 최대한 데이터를 많이 모으는 전수조사**라고 볼 수 있다. 그러나 샘플링을 하는 것이 아니기 때문에 잘못된 결과(편향된 결과)를 줄 수 있다.

Ⅰ. THEORY

빅데이터가 만들어낸 본질적인 변화

"질보단 양으로"

얼마나 커야 빅데이터라고 말할 수 있을까? 데이터가 단순히 크기만 크다고 해서 빅데이터는 아니지만 흔히 빅데이터를 연구하는 사람들은 다음 4가지 조건을 만족하면 규모로써 빅데이터를 만족한다고 말한다.

1. 데이터가 특정 용량 크기 이상이 되었을 때(1GB 이상, 1TB 이상, 1PB 이상)
2. 데이터가 컴퓨터 한 대에서 처리할 수 없을 때
3. 데이터를 Excel에서 열어볼 수 없을 때
4. 데이터의 샘플 수가 많을 때(Image 백 만장 이상, 1000명 이상의 유전체 데이터)

Ⅰ. THEORY

빅데이터가 만들어낸 본질적인 변화

“인과관계에서 상관관계로”

인관관계는 한 변인인 다른 변인의 원인이 되어진다고 믿어지는 관계를 말하며, **상관관계는 여러 변수가 공변하는 함수관계**를 말한다. 인과관계는 변수 발생의 시간 순서는 관계가 없지만 인과관계는 원인 변수는 반드시 결과 변수보다 먼저 발생해야 한다.

예를 들어 키와 몸무게는 키가 커진 다음 몸무게가 커지거나 몸무게가 커진 다음 키가 커지는 것이 아니나 키가 큰 경우 몸무게도 크다. 따라서 키와 몸무게는 상관관계로 보는 것이 맞다. 그러나 공부 양과 시험 성적은 공부 양이 먼저 발생되고 시험 성적이 발생하므로 인과관계로 보는 것이 맞다.

I . THEORY

기계가 공부를 한다? “머신 러닝”

머신 러닝(MACHINE LEARNING)이라는 영어단어를 그대로 직역하면 ‘기계학습’이라는 말이 된다. 인간이 하나부터 열까지 기계에게 가르치는 것이 아니라 기계에게 학습할 거리를 던져주면 이걸 가지고 **스스로 학습하는 기계**를 의미한다. 그럼 기계를 왜 학습시킬까? 사람이 할 수 있는(*혹은 할 수 없는*) 작업을 기계가 할 수 있도록 만들어서, 해당 작업을 아주 빠른 속도로 365일 24시간 내내 자동으로 수행하도록 하기 위함이다. 어떻게 보면 특정 작업에 한해서는 사람보다 일을 훨씬 더 잘 하는 자그마한 AI를 탑재하는 과정으로 볼 수 있는데, 그러한 맥락에서 혹자들은 머신러닝을 약한 인공지능(weak AI)을 위한 기술이라고도 부른다.

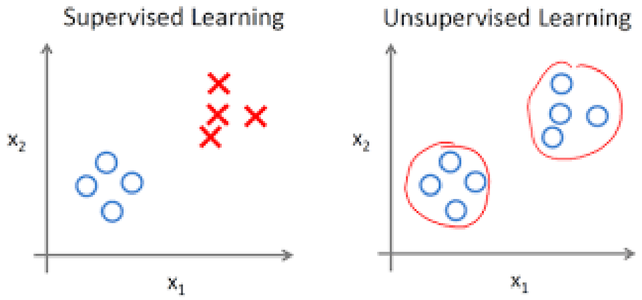
전통적인 통계학에서 회귀 모형식을 계산할 때 최소제곱법 / 최대가능도추정법 등의 방법으로 오차가 최소가 되는 회귀계수를 추정한다. 빅데이터에서 데이터를 받아 최적의 모형을 찾는 과정을 “**머신러닝**” 이라고 한다. 또한 데이터를 읽어 들이고 알맞은 계산을 하는 과정을 “**학습**”한다고 한다.

Ⅰ. THEORY

기계가 공부를 한다? "머신 러닝"

우리가 통계학에서 두 집단의 평균 차이를 검정할 때 "t검정", 세 집단 이상의 평균 차이를 검정할 때 "분산분석" 빈도분석에서의 "독립성 검정"과 "동일성 검정", 인과관계 파악을 위한 "회귀분석" 등 여러 분석기법을 사용한다.

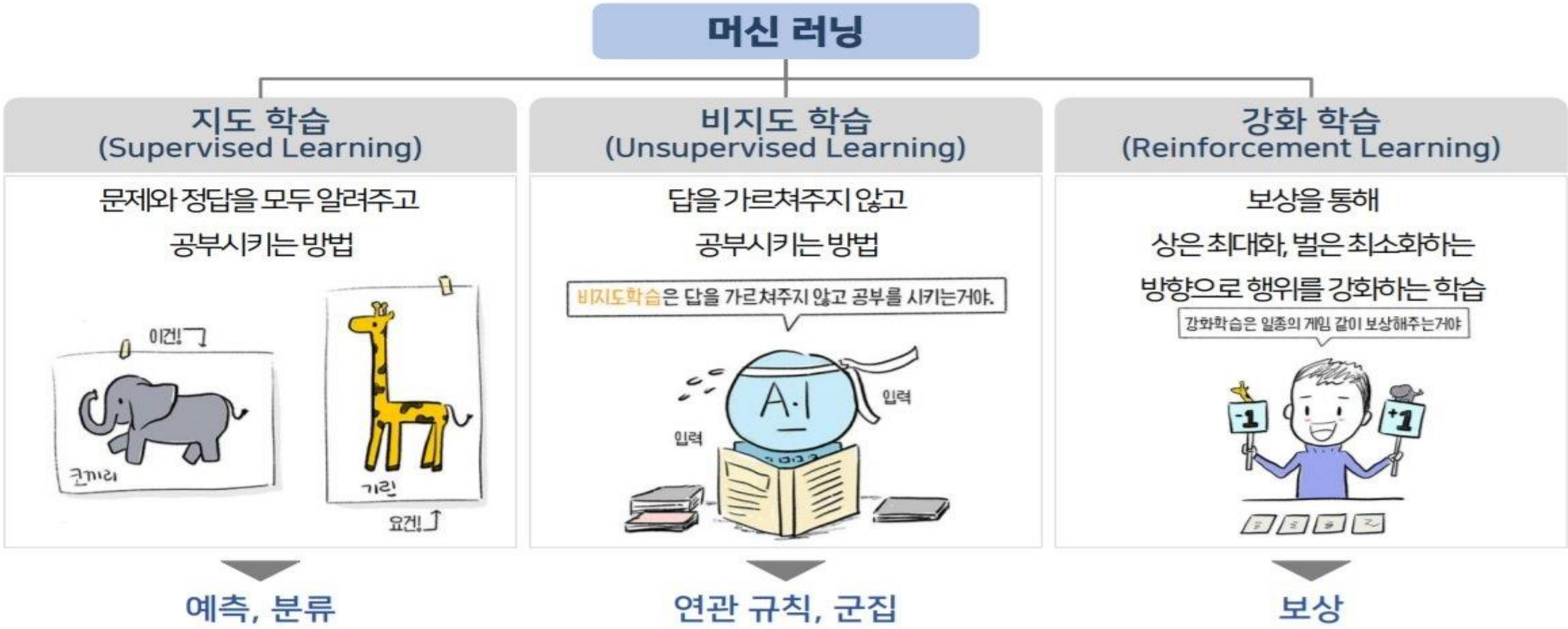
머신 러닝에서도 상황에 맞게 여러 알고리즘들을 사용하여 학습시킨다. 크게 4가지 상황에서 그에 맞는 학습 알고리즘을 다양하게 사용하여 최적의 결과를 나타낸다.



1. **지도학습** : 문제와 정답을 모두 알려주고 공부시키는 방법 ex) 회귀, 분류
2. **비지도학습** : 문제만 알려주고 공부시키는 방법 ex) 군집, 감정 분석
3. **반지도학습** : 일부부만 문제에 대한 답을 알려주고 공부시키는 방법
4. **강화학습** : 보상을 통해 상을 최대로 하고, 벌은 최소화하는 방향으로 공부시키는 방법 ex) 알파고, 시뮬레이션

I . THEORY

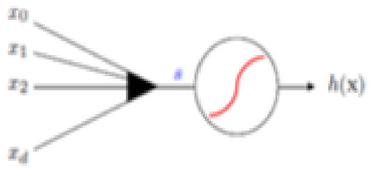
기계가 공부를 한다? “머신 러닝”



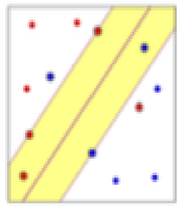
I . THEORY

기계가 공부를 한다? "머신 러닝"

Linear models

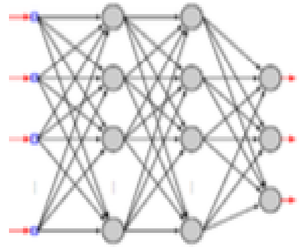


logistic regression

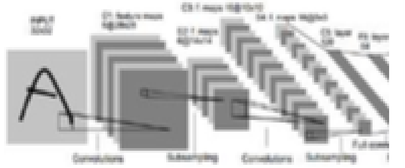


support vector machines

Neural networks

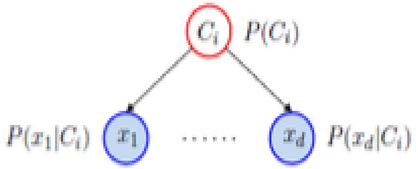


Artificial neural network

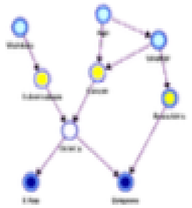


Convolutional neural network

Bayesian models

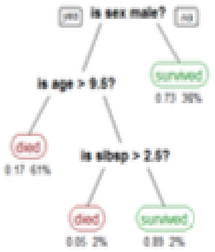


naïve Bayes

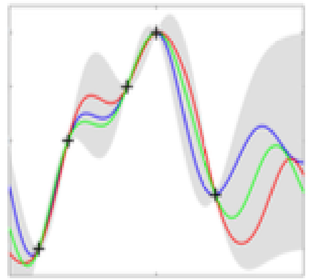


Bayesian network

Nonpara-metric models



Decision tree(C4.5)

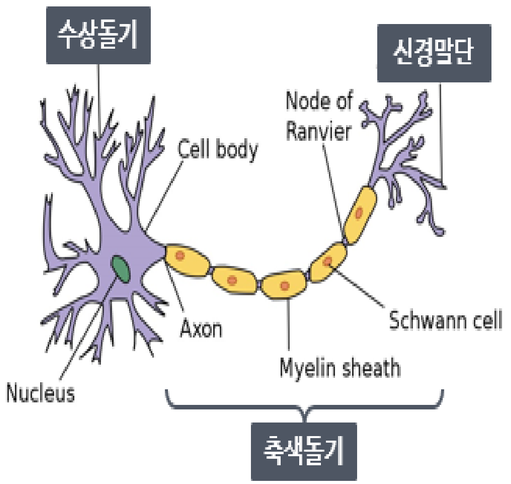


Gaussian process

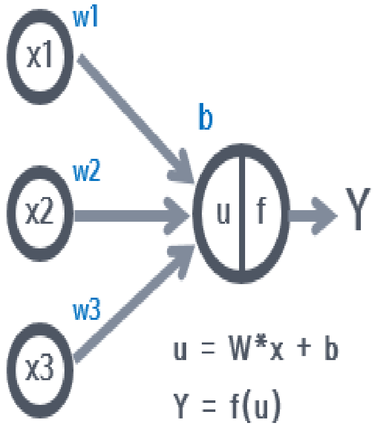
I . THEORY

인간의 뇌를 본 따다. “딥러닝”

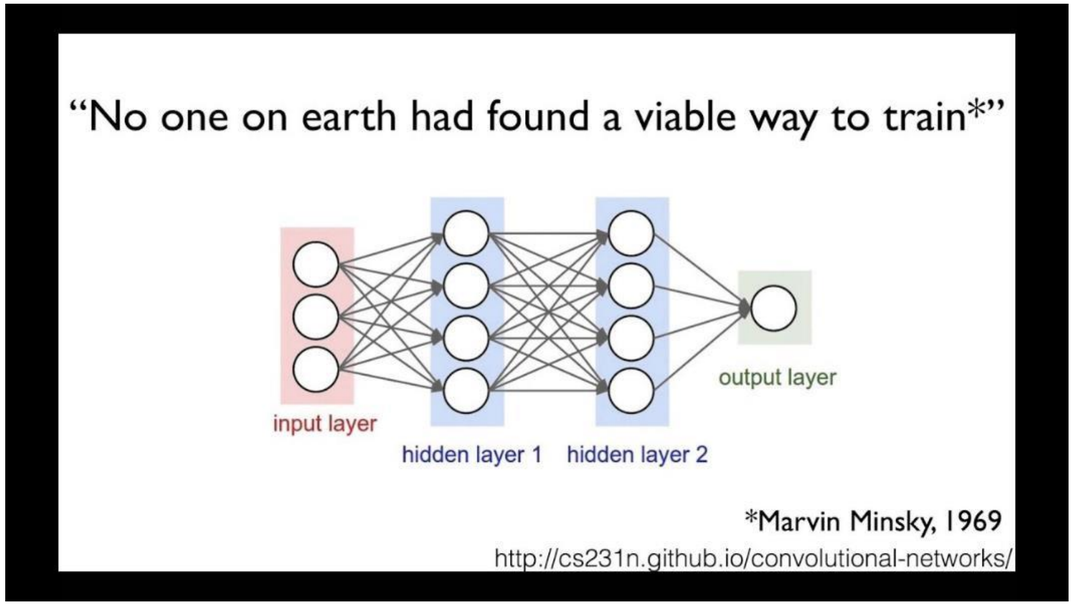
딥 러닝은 머신 러닝의 일종으로 학습 알고리즘을 인간의 뇌를 본떠 만든 알고리즘을 사용한 기계학습이다.
대표적으로 NN, ANN, CNN, RNN 등의 다양한 알고리즘들이 있다.



뉴런의 구조



인공 신경의 구조



I . THEORY

인간의 뇌를 본 따다. “딥러닝”

심층신경망(DNN, Deep Neural Network)

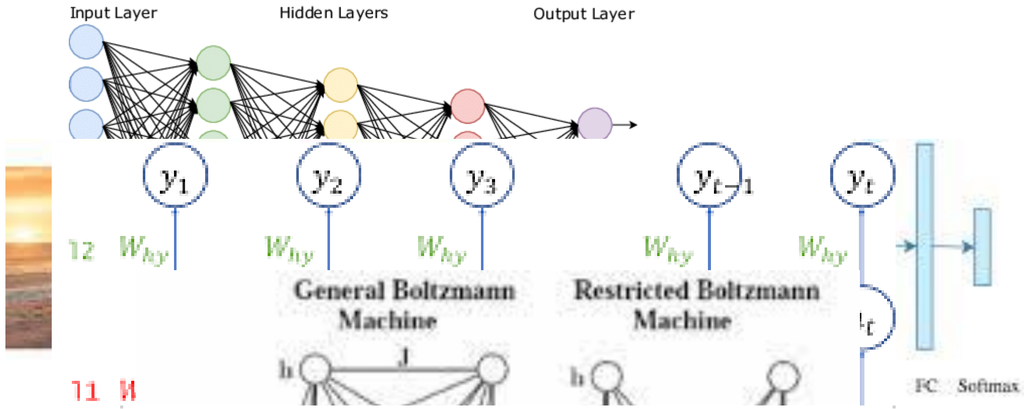
합성곱신경망(CNN, Convolutional Neural Network)

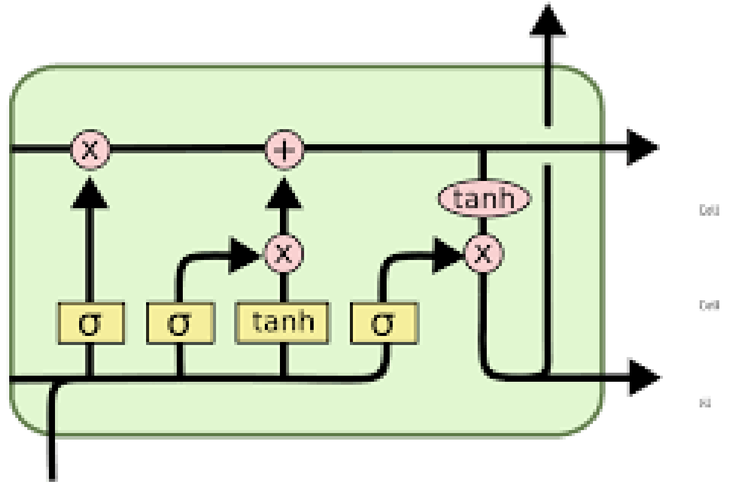
순환신경망(RNN, Recurrent Neural Network)

제한 볼츠만 머신(RBM, Restricted Boltzmann Machine)

심층 신뢰 신경망(DBN, Deep Belief Network)

LSTM(Long Short-Term Memory)





각자 주로 활용되는 분야가 다르다.

INDEX

II . dplyr

데이터 처리의 새로운 강자 “dplyr”

데이터 분석에서 **가장 많은 시간을 차지하는 것은 데이터를 분석에 필요한 형태로 만드는 데이터 전처리 과정**이다. 우리가 공부하면서 보게 되는 책에 있는 예제는 말 그대로 예제일 뿐이지 실제 데이터 분석 업무에서는 바로 모델링이나 시각화에 적합한 형태의 데이터를 얻기 위해서는 지루하고 복잡한 과정을 거치게 된다. 데이터 분석 프로젝트에 걸리는 시간의 절반 이상은 **데이터의 전처리, 변환, 필터링이 차지**하게 되는 것이 보통이다.

R 언어 자체에도 데이터 전처리를 위한 많은 함수가 포함되어 있다. 여기에 각종 패키지의 도움을 받는다면 더욱 쉽고 빠르게 전처리 과정을 마칠 수가 있다. 최근 dplyr이라는 패키지가 새로 발표되어 인터넷상에서 좋은 평가를 받고 있다.

II . dplyr

데이터 처리의 새로운 강자 “dplyr”

```
# dplyr 설치
> install.packages("dplyr")

# dplyr 라이브러리 장착
> library(dplyr)

# dplyr 가 포함되어 있는 종합 패키지 “tidyverse”
> install.packages(“tidyverse")
> library(tidyverse)
```

II . dplyr

데이터 처리의 새로운 강자 “dplyr”

dplyr 주요 함수 및 기능

함수명	내용	유사함수
Filter()	지정한 조건식에 맞는 데이터 추출	Subset()
Select()	열의 추출	data[, c(“Year”, “Month”)]
Mutate()	열 추가	transform()
Arrange()	정렬	order(), sort()
Summarise()	집계	aggregate()

II . dplyr

연습 데이터

 # car 패키지 설치

```
> install.packages("car")
```

 # car 라이브러리 장착

```
> library(car)
```

 # 사용할 데이터

```
> str(UN)
```

```
> str(UN)
'data.frame':       213 obs. of  7 variables:
 $ region        : Factor w/ 8 levels "Africa","Asia",..: 2 4 1 NA 1 3 5 ...
 $ group         : Factor w/ 3 levels "oecd","other",..: 2 2 3 NA 3 2 2 d...
 $ fertility     : num  5.97 1.52 2.14 NA 5.13 ...
 $ ppgdp         : num  499 3677 4473 NA 4322 ...
 $ lifeExpF      : num  49.5 80.4 75 NA 53.2 ...
 $ pctUrban      : num  23 53 67 NA 59 100 93 64 47 89 ...
 $ infantMortality: num  124.5 16.6 21.5 11.3 96.2 ...d
```

II . dplyr

연습 데이터

```
# 데이터의 rowname을 변수로 바꾸기
> un <- rownames_to_column(UN, var="Country")

> head(UN, n=3)
                region     group    fertility    ppgdp   lifeExpF   pctUrban infantMortality
Afghanistan     Asia       other    5.968        499.0   49.49      23          124.535
Albania         Europe     other    1.525        3677.2  80.40      53          16.561
Algeria         Africa     africa   2.142        4473.0  75.00      67          21.458
> head(un, n=3)
      Country       region     group    fertility    ppgdp   lifeExpF   pctUrban   infantMortality
1 Afghanistan       Asia       other    5.968        499.0   49.49      23          124.535
2     Albania       Europe     other    1.525        3677.2  80.40      53          16.561
3     Algeria       Africa     africa   2.142        4473.0  75.00      67          21.458
```

II . dplyr

조건에 맞는 데이터 추출 “filter()”

```
# filter() 기본 사용법
> filter(data, 조건1, 조건2,…)

# un 데이터에서 ppgdp가 80,000 이상인 국가만 출력하여라.
> filter(un, ppgdp > 80000)

# un 데이터에서 group이 oecd인 국가만 출력하여라.
> filter(un, group == "oecd")

# un 데이터에서 ppgdp가 80,000 이상이고, group이 oecd인 국가만 출력하여라.
> filter(un, ppgdp > 80000 & group == "oecd")
```

II . dplyr

조건에 맞는 데이터 추출 “filter()”

 # region이 Asia이고, group은 other이고, ppgdp는 30000 이하 이면서, lifeExpF가 80 이상이고, pctUrban은 80 이상인 국가만 출력하여라.

```
> filter(un, region=="Asia", group=="other",
                ppgdp <= 30000, lifeExpF >= 80, pctUrban >= 80)
```

	Country	region	group	fertility	ppgdp	lifeExpF	pctUrban	infantMortality
1	Republic of Korea	Asia	other	1.389	21052.2	83.95	83	3.647

II . dplyr

데이터 정렬 “arrange()”

 # arrange() 기본 사용법

> arrange(data, 정렬기준 변수1, 정렬기준 변수2,…)

 # un 데이터에서 ppgdp를 오름차순으로 정렬하라.

> arrange(un, ppgdp)

 # un 데이터에서 ppgdp로 오름차순으로 정렬한 다음 동점일 경우 pctUrban의 오름차순으로 정렬하라.

> arrange(un, ppgdp, pctUrban)

 # un 데이터에서 ppgdp로 오름차순으로 정렬한 다음 동점일 경우 pctUrban의 내림차순으로 정렬하라.

> arrange(un, ppgdp, desc(pctUrban))

II . dplyr

데이터 변수 이름 바꾸기 “rename()”

 # rename() 기본 사용법

> arrange(data, 새로운 변수 이름 = 기존의 변수 이름)

 # un 데이터에서 ppgdp의 변수 이름을 new_ppgdp로 변경하여라.

> rename(un, new_ppgdp=ppgdp)

II . dplyr

새로운 변수 생성하기 “mutate()”

 # mutate() 기본 사용법

> mutate(data, 생성할 변수 이름 = 함수식)

 # un 데이터에서 ppgdp를 최소가 0이고 최대가 1인 범위로 새롭게 바꾸어서 new_ppgdp로 생성하라.

```
> mutate(un,new_ppgdp=(ppgdp-min(ppgdp,na.rm=T))/(max(ppgdp,na.rm=T)
                                                  -min(ppgdp,na.rm=T)))
```

max(), min(), mean(), sd() 함수에서 na.rm=TURE 옵션을 주면 결측값을 제외하고 계산해준다.

표준화 : $\frac{\text{값} - \text{최솟값}}{\text{최댓값} - \text{최솟값}}$ 을 하게 되면 값의 범위가 0 ~ 1로 한정된다.

II . dplyr

변수 추출 “select()”

 # select() 기본 사용법

> select(data, 출력할 변수1, 출력할 변수2,…)

 # un 데이터에서 Country와 ppgdp 변수만 출력하여라.

> select(un, Country, ppgdp)

변수명 앞에 “-”를 붙이면 그 변수만 제외되고 출력된다.

변수를 쓰는 순서대로 출력된다.

II . dplyr

그룹핑 함수 “group_by()”와 집계 함수 “summarise()”

 # group_by() 기본 사용법

> group_by(data, 그룹핑할 기준 변수)

 # summarise() 기본 사용법

> summarise(data, 집계할 함수)

 # group 별로 ppgdp의 평균값을 구하라. 단 NA 값은 제거하여라.

> summarise(group_by(filter(un,!is.na(group)),group),mean(ppgdp,na.rm=T))

filter(un,!is.na(group)) : un 데이터에서 group 이 na인 값을 제거 하고 fiter한다.

INDEX

III . 데이터 마이닝

데이터 마이닝

 # 데이터 마이닝이란..?

대규모로 저장된 데이터 안에서 체계적이고 자동적으로 통계적 **규칙이나 패턴**을 찾아 내는 것이다. 다른 말로는 KDD(데이터베이스 속의 지식 발견: Knowledge-discovery in databases)라고도 일컫는다.

Linear Regression

Logistic Regression

Decision Tree

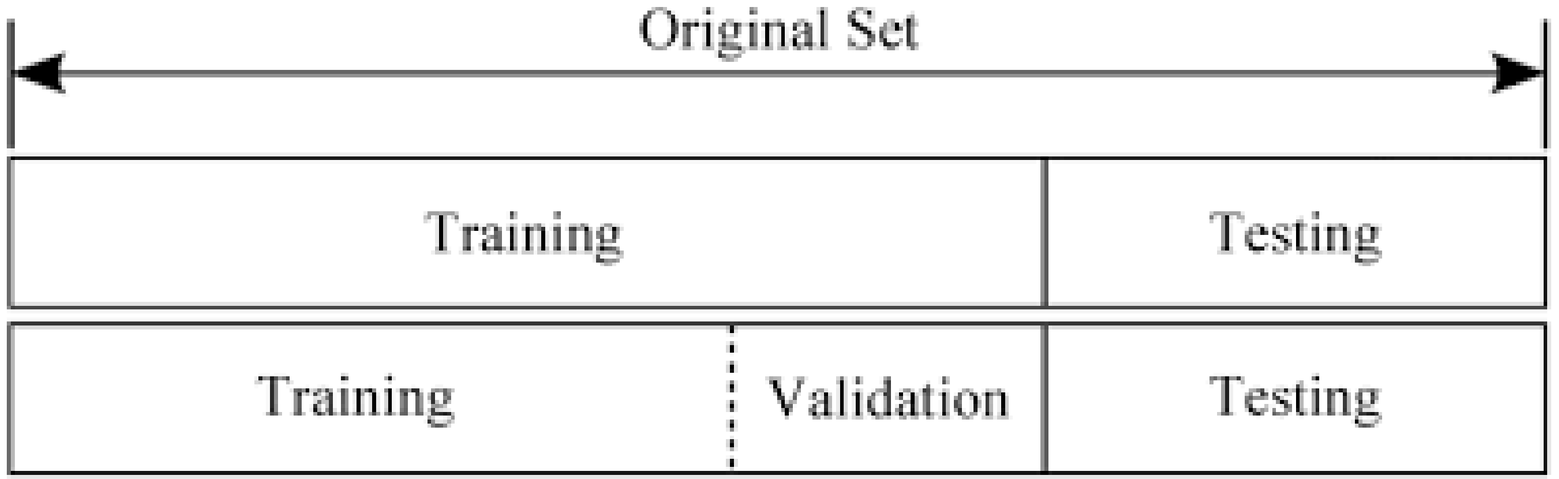
Neural Network

III . 데이터 마이닝

데이터 준비

데이터 마이닝을 위한 데이터 분할

일반적으로 데이터 마이닝에서는 데이터를 3가지로 분류한다. 모형을 학습하기 위한 Train Set, 만든 모형을 검증하고 보완하기 위한 Validation Set, 실제 모형을 적용해야 할 미지의 데이터 Test Set으로 분류한다.



III . 데이터 마이닝

모델 준비 및 선택

 # 하이퍼 파라미터(Hyper Parameter)

학습을 시켜야 할 변수가 아니라 사람들이 선험적 지식으로 설정을 하거나 또는 외부 모델 메커니즘을 통해 자동으로 설정이 되는 변수를 말한다.

Linear Regression : α

Logistic Regression : Link Function, 분류 기준 값

Decision Tree : Node 수

Neural Network : Hidden Layer 수

III . 데이터 마이닝

모델 평가

Linear Regression

일반적으로 회귀 모형을 평가할 때는 AIC, BIC, 결정 계수, 수정된 결정 계수, Cp 통계량, MSE, RMSE, MAPE 등을 기준으로 평가한다.

Logistic Regression, Decision Tree, Neural Net

분류 분석은 종속변수의 값이 0 또는 1로 나오므로 일반적인 회귀 분석에 사용하는 평가값을 사용하기에는 다소 무리가 있다. 따라서 로지스틱 모형에서는 오분류표, 특이도, 민감도, F1-Score, ROC Curve, AUC 등을 사용한다.

III . 데이터 마이닝

모델평가

오분류표

2X2 행렬로 표현되며, 일반적으로 행을 실제 값, 열을 예측값으로 놓는다.

		Predicted Data		Total
		Predicted Condition POSITIVE	Predicted Condition NEGATIVE	
Acutal Data	Condition TRUE	TP True Positive	FN False Negative	P
	Condition FALSE	FP False Positive	TN True Negative	N
Total		P`	N`	P+N

III . 데이터 마이닝

모델평가

		Predicted Data		Total
		Predicted Condition POSITIVE	Predicted Condition NEGATIVE	
Acutal Data	Condition TRUE	TP True Positive	FN False Negative	P
	Condition FALSE	FP False Positive	TN True Negative	N
Total		P`	N`	P+N

정분류율(ccr, Correcet Classfication Rate)

전체 관측치 중 실제값과 예측값이 일치하는 정도

$$\text{Accuarcy} = \frac{TP+TN}{TP+FP+FN+TN}$$

오분류율

모형이 제대로 예측하지 못한 관측치

$$\text{Error rate} = \frac{FP+FN}{TP+FP+FN+TN} = 1 - Acc$$

III . 데이터 마이닝

모델평가

		Predicted Data		Total
		Predicted Condition POSITIVE	Predicted Condition NEGATIVE	
Acutal Data	Condition TRUE	TP True Positive	FN False Negative	P
	Condition FALSE	FP False Positive	TN True Negative	N
Total		P`	N`	P+N

민감도

실제 True인 관측치 중 예측이 True인 관측치

$$\text{Sensitivity} = \frac{TP}{TP+FN}$$

특이도

실제 False인 값이 예측에서도 False인 관측치

$$\text{Specificity} = \frac{TN}{FP+TN}$$

III . 데이터 마이닝

모델평가

		Predicted Data		Total
		Predicted Condition POSITIVE	Predicted Condition NEGATIVE	
Acutal Data	Condition TRUE	TP True Positive	FN False Negative	P
	Condition FALSE	FP False Positive	TN True Negative	N
Total		P`	N`	P+N

 # 정확도

True로 예측한 관측치 중 실제 True인 관측치

$$\text{precision} = \frac{TP}{TP+FP}$$

 # 재현율

실제 True인 값 중 예측치가 적중한 정도

민감도와 동일한 지표

$$\text{recall} = \frac{TP}{TP+FN}$$

III . 데이터 마이닝

모델평가

		Predicted Data		Total
		Predicted Condition POSITIVE	Predicted Condition NEGATIVE	
Acutal Data	Condition TRUE	TP True Positive	FN False Negative	P
	Condition FALSE	FP False Positive	TN True Negative	N
Total		P`	N`	P+N

F1 – Score

정확도와 재현율의 조화평균

$$F1 = \frac{2\ X\ precision\ X\ recall}{Precision+recall}$$

III . 데이터 마이닝

모델평가

ROC Curve

Receiver Operating Characteristic curve로 , FRR과 TPR을 각각 x, y축으로 놓은 그래프이다. TPR은 True Positive Rate로 민감도를 의미한다. FPR은 False Positive Rate로 1-특이도를 의미한다.

TPR = 1인 케이스를 1로 올바르게 예측한 비율
FPR = 0인 케이스를 1로 잘못 예측한 비율

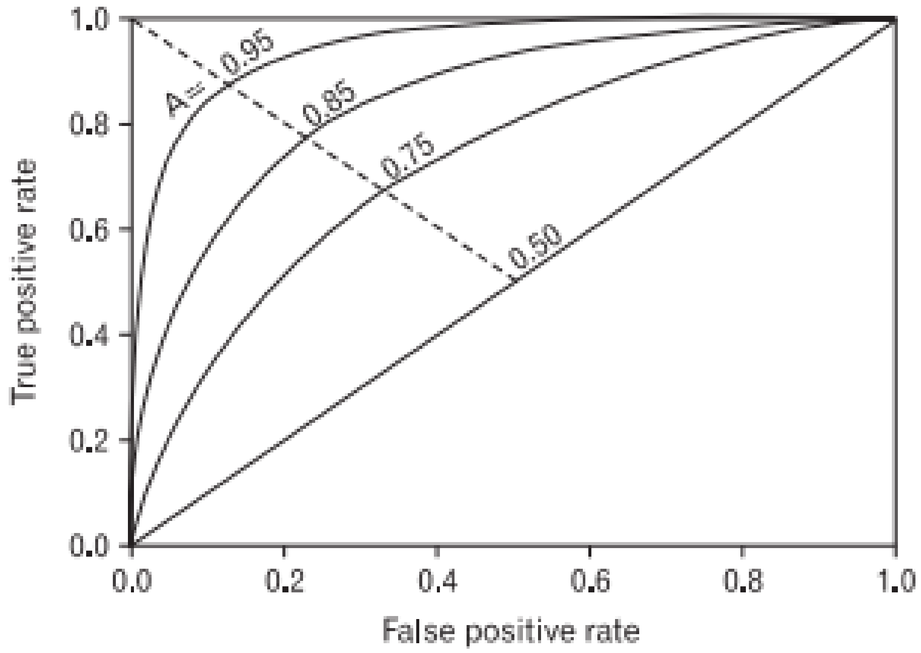
Trade off

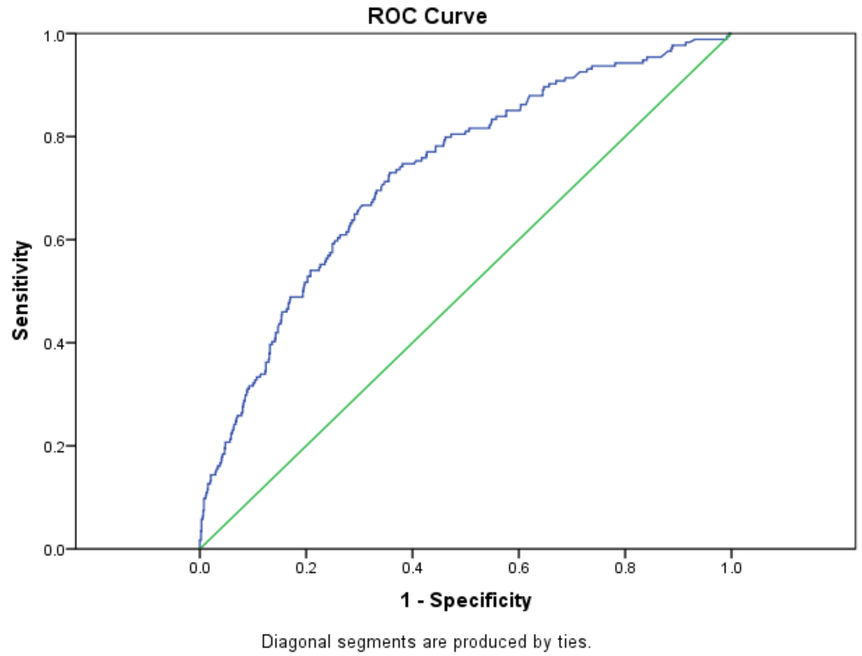
TPR과 FPR은 서로 반비례 관계이다. 그러나 특이도와 민감도 중 어느 것이 좋다고 말하기 어렵다. 따라서 이 둘의 그래프인 ROC Curve를 그려서 종합적으로 판단해야 올바른 모형을 선택할 수 있다.

III . 데이터 마이닝

모델평가

ROC Curve



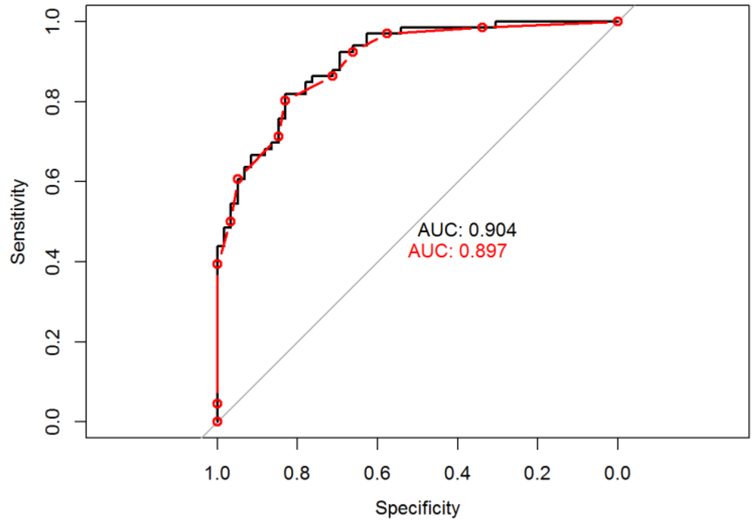


III . 데이터 마이닝

모델평가

AUC

Area Under Curve 로 ROC Curve의 밑의 면적을 나타내는 값이다. 일반적으로 0.5 에서 1사이의 값을 가지며, 1에 가까울수록 좋은 모형이라고 평가한다.



III . 데이터 마이닝

모델평가

수정된 ccr

정분류율(ccr)에서 단순 추측으로 보장된 정분류율을 제거한 정분류율이다. 모형에 의해 분류 시켰을 때, 단순 추측에 의한 분류보다 얼마나 오류를 감소시킬 수 있을지 측정한 값이다.

		Prediction		Total
		$\hat{Y} = 1$	$\hat{Y} = 0$	
Observation	$Y = 1$	n_{11}	n_{10}	n_{1+}
	$Y = 0$	n_{01}	n_{00}	n_{0+}
Total		n_{+1}	n_{+0}	n

수정된 ccr

$$= \frac{n_{11} + n_{00} - \max(n_{r+})}{n} \times 100$$

III . 데이터 마이닝

Over Fitting

 # 과대 적합이란..?

Train data의 정확도가 95%가 나오고, Test data의 정확도가 80%가 나왔다면 좋은 모델일까?

이는 모형이 Train data에 과대 적합 되었다고 말한다. 과대 적합은 Train data에 너무 깊숙하게 피팅되어 새로운 New data에 대하여 반응도가 떨어져 예측력이 급격하게 낮아지는 현상을 말한다. 과대 적합은 여러 원인에 의해 일어나는데 Test data가 Train data에 비해 현저하게 적거나, Decision Tree에서 노드 수를 너무 많이 설정하거나, Neural Network에서 Hidden layer 수를 너무 많이 설정 하는 경우 발생한다.

따라서 과대 적합을 맞기 위해 적절한 데이터 분리와 최적의 Hyper Parameter 설정이 필요하다.

INDEX

Ⅳ . SQL

데이터베이스

데이터베이스란..?

여러 사람들이 공유하고 사용할 목적으로 통합 관리되는 정보의 집합이다.

관계형 데이터베이스란..?

우리가 흔히 접하고 다루는 2차원 테이블로 표현 된 데이터베이스를 뜻한다. 테이블은 유일한 이름을 가지며, 테이블의 각 열은 필드(filed), 행은(record)라고 부른다.

속성과 도메인..?

속성은 필드와 열이 교차되는 지점에 들어가 있는 값 자체를 말하며, 도메인은 각 필드의 자료 형태를 말한다.
ex) 속성명 : 학번 / 도메인 : int　　속성명 : 학점 / 도메인 : float

Ⅳ . SQL

데이터베이스

관계형 데이터베이스에서 키(key)의 역할은..?

테이블에서 레코드(행)을 유일하게 식별할 수있는 속성들의 집합을 “키” 또는 “슈퍼키”라고 한다. 이 키에는 반드시 널값(null value)는 안된다. ex) 주민등록번호, 학번, 전화번호 등등…

후보키, 기본키, 외래키란..?

후보키는 키가 될 수 있는 모든 키를 말하며, 기본키는 후보키들 중에서 사용자가 키로 선택한 키를 말하며, 외래키는 어떤 테이블의 기본키를 다른 테이블에서 사용할 때 지정하는 키다.

Ⅳ . SQL

SQL

SQL이란..?

SQL은 관계형 데이터베이스 관리 시스템(RDBMS)의 데이터를 관리하기 위해 설계된 특수 목적의 프로그래밍 언어이다. 관계형 데이터베이스 관리 시스템에서 자료의 검색과 관리, 데이터베이스 스키마 생성과 수정, 데이터베이스 객체 접근 조정 관리를 위해 고안되었다. 많은 수의 데이터베이스 관련 프로그램들이 SQL을 표준으로 채택하고 있다.

현재 널리 사용되는 SQL

MySQL, PostgreSQL, 파이썬, PHP, 자바 등등

Ⅳ . SQL

SAS에서 PROC SQL을 위한 준비 단계

 # 데이터 준비

lms 가서 데이터 다운 받아 오자^^

 # 가상 라이브러리 생성

```
libname sql "C:₩Users₩user₩Desktop₩University₩빅데이터분석₩Data";
```

현재 데이터가 있는 파일을 가상 라이브러리로 지정해준다.
R에서 read.csv() 등등 사용해서 라이브러리에 올려놓는 것과 같은 논리이다.

libname 뒤에는 가상 라이브러리의 이름으로 아무거나 작성한다.

Ⅳ . SQL

PROC SQL

PROC SQL의 일반적인 형태

```
proc sql;
title 'Population of Large Countries Grouped by Continent';
Select Continent, sum(Population) as TotPop format=comma15.
From sql.countries
Where Population gt 1000000
group by Continent
Order by TotPop;
quit;
```

나 이제부터 주문할거야!
제목은..?
무슨 변수 볼 거야..?
무슨 데이터에서..?
보고 싶은 조건은..?
그룹핑 할 거야..?
정렬할 거야..?
내 주문 실행해줘!

Ⅳ . SQL

PROC SQL

```
proc sql;
select *
from sql.countries;
quit;
```

select는 출력할 변수를 지정한다. 단 SQL 문에서 '*' 을 작성하면 모든 변수를 출력한다.

from 은 어떤 데이터에서 작업을 할 건지 지정한다. 일반적으로 "라이브러리명.데이터이름" 으로 작성한다.

Ⅳ . SQL

PROC SQL

```
proc sql;
select *
from sql.countries
where Population gt 5000000;
quit;
```

참고) gt는 greater than의 약자로
'초과'의 의미를 나타내며
">"를 사용해도 같은 결과가 출력된다.

모든 변수를 출력하되, 데이터는 sql 라이브러리에 있는 countries 를 사용하고,
Population 변수가 5,000,000 명 이상인 국가만 출력하라.

Ⅳ . SQL

PROC SQL

```
proc sql;
select *
from sql.countries
where Population gt 5000000
order by Population desc;
quit;
```

모든 변수를 출력하되, 데이터는 sql 라이브러리에 있는 countries 를 사용하고,
Population 변수가 5,000,000 명 이상인 국가만 Population 변수를 기준으로 내림차순 정렬하여라.

IV . SQL

PROC SQL

```
proc sql;
select Continent, sum(Population)
from sql.countries
group by Continent
order by Continent;
quit;
```

Continent와, Population의 합만 출력하되, 데이터는 sql 라이브러리에 있는 countries 를 사용하고,
Continent를 기준으로 그룹핑하고 Continent 기준으로 오름차순으로 출력하여라.

Ⅳ . SQL

PROC SQL

```
proc sql ;
title 'U.S. Cities with Their States and Coordinates';
select City, State
from sql.uscitycoords;
quit ;
```

출력의 제목을 “U.S. Cities with Their States and Coordinates” 으로 하고, City와 State 변수만 출력하되, 데이터는 sql 라이브러리에 있는 uscitycoords를 사용하여라.

Ⅳ . SQL

PROC SQL

```
proc sql;
select distinct Continent
from sql.unitedstates;
quit ;

proc sql;
select Continent
from sql.unitedstates;
quit ;
```

참고) SELCT 에서 변수 앞에 distinct를 적으면 중복은 제외시켜준다.

IV . SQL

PROC SQL

```
proc sql;
describe table sql.unitedstates;
quit ;

# describe table 은 데이터 개요로 R에서 str() 함수와 같은 역할이다.
```

Ⅳ . SQL

PROC SQL

```
proc sql outobs=12;
title 'U.S. Postal Codes';
select 'Postal code for', Name, 'is', Code
from sql.postalcodes;
quit ;
```

select 문에서 총 4개의 변수를 출력하는 것으로 지정('Postal code for', Name, 'is', Code)

'Postal code for', 'is' 는 모든 레코드에 모두 같은 문자열로 들어간다.

Ⅳ . SQL

PROC SQL

```
proc sql outobs=12;
title 'Low Temperatures in Celsius';
select City, (AvgLow-32) * 5/9 format=4.1
from sql.worldtemps;
quit ;
```

섭씨 온도 = $\frac{5}{9}$ (화씨 온도 -32)

format=4.1 : 정수의 자릿수가 4자리이고, 소수점은 한 자리까지 표기하여라.

IV . SQL

PROC SQL

```
proc sql outobs=12;
title 'Range of High and Low Temperatures in Celsius';
select City, (AvgHigh - 32) * 5/9 as HighC format=5.1,
       (AvgLow - 32) * 5/9 as LowC format=5.1,
       (calculated HighC-calculated LowC) as Range format=4.1
from sql.worldtemps;
quit;
```

calculated 를 사용하면 새로 만들어진 변수로 또 다른 변수를 만들 수 있다.

as 를 사용하면 새로 만든 변수에 이름을 지정할 수 있다.

IV . SQL

PROC SQL

```
proc sql;
title 'Continental Low Points';
select Name, LowPoint as LowPoint
from sql.continents;
quit ;

# 현재 LowPoint 변수에 결측값이 있는 상태다.
```

Ⅳ . SQL

PROC SQL

```
proc sql;
title 'Continental Low Points';
select Name, coalesce(LowPoint, 'Not Available') as LowPoint
from sql.continents;
quit ;
```

coalesce()를 사용하여 결측값을 “'Not Available” 로 대체한다.

IV . SQL

PROC SQL

```
proc sql outobs=12;
title 'Climate Zones of World Cities';
select City, Country, Latitude,
case
   when Latitude gt 67 then 'North Frigid'
   when 67 ge Latitude ge 23 then 'North Temperate'
   when 23 gt Latitude gt -23 then 'Torrid'
   when -23 ge Latitude ge -67 then 'South Temperate'
   else 'South Frigid'
end as ClimateZone
from sql.worldcitycoords
order by City;
quit ;
```

Latitude 라는 변수를 생성

등급으로 만들 예정

# gt = greater than
  ge = greater equal

gt, ge의 기준은 무조건 왼쪽!

else는 마지막 구간을 처리

end as는 결측값을 처리

IV . SQL

PROC SQL

```
proc sql outobs=12;
title 'Assigning Regions to Continents';
select Name label='State', Area format=comma10.
from sql.unitedstates;
quit;
```

```
proc sql outobs=12;
title 'Assigning Regions to Continents';
select Name label='State', Area format=comma5.
from sql.unitedstates;
quit;
```

출력결과 주의!!!

Ⅳ . SQL

PROC SQL

```
proc sql;
title 'Countries with Missing Continents';
select Name, Continent
from sql.countries
where Continent is missing;
quit;
```

Continent 가 결측값인 데이터 출력

Ⅳ . SQL

PROC SQL

```
proc sql outobs=12;
title 'Equatorial Cities of the World';
select City, Country, Latitude
from sql.worldcitycoords
where Latitude between -5 and 5;
quit;
```

“between” 과 “and” 를 사용해 조건을 범위로 줄 수 있다.

IV . SQL

PROC SQL

```
proc sql;
title1 'Country Names that Begin with the Letter "Z"';
title2 'or Are 5 Characters Long and End with the Letter "a"';
select Name
from sql.countries
where Name like 'Z%' or Name like '____a';
quit;
```

Name이 “Z” 로 시작하거나 “____a” 의 형태인 데이터만 출력

Ⅳ . SQL

PROC SQL

```
proc sql;
title '"New" U.S. States';
select Name
from sql.unitedstates
where Name EQT 'New ';
quit;
```

eqt는 문자열을 비교해서 잘린 문자열이 같으면 출력한다.

Ⅳ . SQL

PROC SQL

Symbol	Definition	정의
EQT	Equal to truncated strings	코드가 같으면 출력
GTT	Greater than truncated strings	코드가 크면 출력
LTT	Less than truncated strings	코드가 작으면 출력
GET	Greater than or equal truncated strings	코트가 크거나 같으면 출력
LET	Less than or equal truncated strings	코드가 작거나 같으면 출력
NET	Not equal to truncated strings	코드가 다르면 출력

Ⅳ . SQL

PROC SQL

```
proc sql outobs=12;
title 'World Features with a Depth of Less than 500 Feet';
select Name, Depth
from sql.features
where Depth lt 500
order by Depth;
quit;
```

```
proc sql outobs=12;
title 'World Features with a Depth of Less than 500 Feet';
select Name, Depth
from sql.features
where Depth lt 500 and Depth is not missing
order by Depth;
quit ;
```

R에서 na.rm=T와 같은 역할

Ⅳ . SQL

PROC SQL

Function	Definition	Function	Definition
AVG, MEAN	평균 계산	STDERR	평균의 표준 오차
COUNT, FREQ, N	빈도 계산	SUM	합계
CSS	제곱합	SUMWGT	가중 합계
CV	변동계수	USS	결정 계수
MAX(MIN)	최댓값(최솟값)	VAR	분산
NMISS	결측값 수	T	T검정 통계량 값
RANGE	범위		
STD	표준편차		

IV . SQL

PROC SQL

다음의 출력결과가 나오게 프로그래밍을 코딩하시오.

데이터는 countries를 사용

```
proc sql outobs=12;
title 'Percentage of World Population in Countries';
select Name, Population format=comma14.,
(Population / sum(Population) * 100) as Percentage
format=comma8.2
from sql.countries
order by Percentage desc;
quit;
```

Percentage of World Population in Countries

Name	Population	Percentage
China	1,202,215,077	21.10
India	929,009,120	16.30
United States	263,294,808	4.62
Indonesia	202,393,859	3.55
Brazil	160,310,357	2.81
Russia	151,089,979	2.65
Bangladesh	126,387,850	2.22
Japan	126,345,434	2.22
Pakistan	123,062,252	2.16
Nigeria	99,062,003	1.74
Mexico	93,114,708	1.63
Germany	81,890,690	1.44

Ⅳ . SQL

PROC SQL

```
proc sql;
title 'Number of Continents in the Countries Table';
select count(distinct Continent) as Count
from sql.countries;
quit;

# 대륙을 중복을 제외하고 몇 개 있는지 출력
```

Ⅳ . SQL

PROC SQL

```
proc sql;
title 'Number of Countries in the Sql.Countries Table';
select count(*) as Number
from sql.countries;
quit;

# 데이터의 행의 개수를 출력
```

INDEX

V . TEST

미안해요..

시험지 못 만들었어요..

시간이 없어서..

THANK YOU